เรา ดอสและดิสค์

บุญเลิศ เอี่ยมทัศนา

การศึกษาเรื่องราวของระบบจัดการโดยเฉพาะระบบจัดการ MS-DOS ซึ่งบนเครื่องไอบีเอ็ม- พีซี เรียกว่า PC-DOS และผมจะขอเรียกสั้นๆ ว่า ดอส น่าจะแบ่งออกได้เป็น 3 หัวข้อคือ การใช้งานดอส การใช้โปรแกรมย่อยในดอส และเรื่องราวภายในของดอส

ทั้ง 3 เรื่องนี้มีอยู่ในคู่มือของดอสคือ Disk Operating System ของ Microsoft แต่ในบ้านเรามีทั้งเล่มเต็มและเล่มไม่เต็ม (เล่มเต็มมีอยู่ 14 บทกับผนวก A ถึง L) แต่ถึงมีเล่มเต็ม ก็อ่านทำความเข้าใจได้ยากพอสมควร และก็มีปัญหาอีกว่าใครควรจะศึกษาเพียงไร จึงจะขอเรียนให้ทราบดังนี้

การใช้งานดอส

สำหรับผู้ที่ใช้งานเครื่องโดยทั่วไป ควรจะศึกษาเฉพาะภาคแรกของคู่มือดอส คือบทที่ 1 ถึง 8 ก็นับว่าเพียงพอ สำหรับภาคสอง คือบทที่ 9 ถึง 14 สำหรับผู้ที่เขียนโปรแกรมที่เกี่ยวข้องกับระบบจัดการโดยเฉพาะผู้เขียนโปรแกรมในภาษาเครื่อง

โปรแกรมย่อยในดอส

การเรียกใช้โปรแกรมย่อยในดอสจะเรียกผ่านซอฟต์แวร์อินเตอร์รัพต์ คือคำสั่ง INT n โดยที่ n เป็นหมายเลขอินเตอร์รัพต์ เรื่องนี้เป็นไปตามแนวความคิดในการออกแบบซีพียู 8086/8088 ซึ่งดอสนำมาใช้แต่ดอสแบ่งออกเป็นการอินเตอร์รัพต์กับการใช้ฟังก์ชันคอลล์ (function call) โดยที่ฟังก์ชันคอลล์ก็คือ INT 21h แต่ดอสกำหนดให้เราให้ค่ารีจิสเตอร์ AH เป็นหมายเลขโปรแกรมย่อยอีกทอดหนึ่ง

ผู้ที่ต้องศึกษาในเรื่องนี้ คือผู้ที่เขียนโปรแกรมในภาษาเครื่อง รองลงไปคือผู้ที่เขียนโปรแกรมในภาษาสูงที่เรียกใช้โปรแกรมย่อยในภาษาเครื่องได้ และมีความจำเป็นที่จะต้องใช้โปรแกรมในภาษาเครื่องสนับสนุนโปรแกรมของตน ซึ่งจะศึกษาได้ในคู่มือดอส ภาคผนวก D แต่คำอธิบายที่ละเอียดกว่าจะอยู่ในหนังสือ Microsoft MS-DOS, Operating System, Programmer's Reference Manual

เรื่องภายในของดอส

เป็นเรื่องที่เกี่ยวข้องกับดิสค์โดยตรง ผู้ที่ควรศึกษาในเรื่องนี้คือ ผู้ที่เขียนโปรแกรมภาษาเครื่อง โดยเฉพาะโปรแกรมที่เกี่ยวพันกับระบบจัดการกับดิสค์ กล่าวไว้ในคู่มือดอส ภาคผนวก B, C และ E แต่ก็อ่านได้ยากพอสมควร ต้องอ่านหนังสือประกอบดังที่กล่าวในเอกสารอ้างอิง ผมจึงจะนำเรื่องราวเหล่านี้มาเขียนเฉพาะที่จำเป็น และจะแยกเป็น 2 หัวข้อ คือ ดอสกับดิสค์ และเรากับดอส และเพื่อให้เรื่องกระชับผมจะหมายถึง ดอสเวอร์ชัน 2.x ซึ่งมีใช้มากในบ้านเรา

ดอสกับดิสค์

เมื่อเราฟอร์แมตดิสค์ทั้งที่มีซิสเท็ม (เช่น FORMAT B:/S) หรือไม่มี (เช่น FORMAT B:) ดอสจะแบ่งพื้นที่ในดิสค์ออกเป็น 40 แทร็ก (track) ในแต่ละแทร็กแบ่งออกเป็น 9 เซกเตอร์ เซกเตอร์ละ 512 ไบต์ทั้ง 2 หน้า ดังนั้นในดิสค์ 1 แผ่น จะมี 2×40×9 = 720 เซกเตอร์ คือ 368, 640 ไบต์

คำว่าเซกเตอร์ที่กล่าวมานี้ เป็นฟิสิคัลเซกเตอร์ (physical sector) ซึ่งจะมีวิธีกำหนดเซกเตอร์เป็น เช่น แทร็ก 0 หน้า 0 เซกเตอร์ 1 ใช้ใน BIOS ซึ่งผมจะไม่นำมากล่าวถึง ที่จะนำมากล่าวถึงคือเซกเตอร์ในดอสเรียกว่าเป็นลอจิคัลเซกเตอร์ (logical sector) ให้เราอ้างถึงเซกเตอร์ต่างๆ เหล่านี้ได้ตั้งแต่เซกเตอร์ 0 ถึงเซกเตอร์ 719 โดยเราไม่ต้องคำนึงถึงว่าจะอยู่ที่ฟิสิคัลเซกเตอร์ใด

เมื่อดอสดำเนินการฟอร์แมตดิสค์จะแบ่งพื้นที่ในดิสค์ออกเป็น 4 ส่วนคือ

● **Boot** อยู่ที่เซกเตอร์ 0 ประกอบด้วยโปรแกรมในการโหลดโปรแกรมในส่วนที่เป็นระบบจัดการของดอสลงหน่วยความจำ ถึงแม้เราจะฟอร์แมตดิสค์ในแบบที่ไม่มีซิสเท็ม ก็มีส่วนที่เป็นบูทนี้ แต่ตัวโปรแกรมไม่เหมือนกัน

● **FAT** (File Allocation Table) ทำหน้าที่เป็นบันทึกของดอสว่า เซกเตอร์ใดได้ใช้งานไปแล้ว คือได้กำหนดให้แก่ไฟล์ใดไปแล้ว โดยเก็บเป็นพอยน์เตอร์ไปยังคลัสเตอร์ (Clusters) โดยที่ดอสรวม 2 เซกเตอร์เป็น 1 คลัสเตอร์ เพื่อประหยัดพื้นที่ในดิสค์ในการเก็บ FAT แต่ถึงกระนั้นก็ดี ยังต้องใช้พื้นที่ในดิสค์ถึง 2 เซกเตอร์ และเพื่อป้องกันการสูญเสีย เพราะ FAT มีความสำคัญมาก จึงได้ทำก๊อบปี้ไว้ด้วย ทำให้ใช้เนื้อที่ในดิสค์ถึง 4 เซกเตอร์ อยู่ที่เซกเตอร์ 1 ถึงเซกเตอร์ 4 (สำหรับรายละเอียดที่มากไปกว่านี้จะของดกล่าวเพราะเห็นว่าเป็นเรื่องที่ไม่จำเป็น เว้นไว้จะต้องการเขียนโปรแกรมกู้ไฟล์เอง)

● **Directory** เปรียบเสมือนสารบัญของดอสอยู่ที่เซกเตอร์ 5 ถึงเซกเตอร์ 11 รวม 7 เซกเตอร์ มีรายละเอียดที่จะกล่าวต่อไป

● **Data** คือส่วนที่เก็บข้อมูลของไฟล์ต่างๆ อยู่ที่เซกเตอร์ 12 ถึงเซกเตอร์ 719 (คำว่า Data ซึ่งในภาษาไทยอาจเรียกทับศัพท์หรือแปลว่า ข้อมูล เป็นคำรวมซึ่งอาจจะหมายถึงข้อมูลแท้ๆ หรือรหัสคำสั่งเครื่องก็ได้

ไดเรคทอรี (Directory)

เมื่อเราเปิดไฟล์ ดอสจะบันทึกชื่อไฟล์และรายละเอียดต่างๆ ไว้ในส่วนที่เป็นไดเรคทอรี สำหรับไฟล์หนึ่งๆ จะใช้เนื้อที่ในดิสค์ 32 ไบต์ เรียกว่าหนึ่งเอนทรี (directory entry) ดังแสดงในรูปที่ 1 และมีรายละเอียดดังนี้

ชื่อไฟล์ (Filename) 8 ไบต์ ตามข้อกำหนดของดอส ซึ่งจะต้องไม่มีวรรค แต่ผู้ใช้อาจตั้งให้มีวรรคก็ได้ และเป็นการพลิกแพลงแบบหนึ่ง เพราะจะใช้กับคำสั่ง DEL และ COPY ไม่ได้ สำหรับชื่อไฟล์ไบต์แรกมีความหมายต่อไปอีกคือ

00h หมายถึง เอนทรีนี้ยังไม่เคยใช้งาน และเนื่องจากดอสใช้เอนทรีแบบเรียงลำดับ เราจึงนำมาใช้เป็นเงื่อนไขว่าจบไดเรคทอรี ดังที่ใช้ในตัวอย่างการเขียนโปรแกรมภาษาเครื่อง

E5h หมายถึง ไฟล์ที่ถูกลบคือ ERASE แต่ความเป็นจริงเป็นการยกเลิก เพราะดอสเพียงแต่ให้ค่า E5h ที่ไบต์นี้เท่านั้น ไม่ได้ลบข้อมูลที่เก็บในไฟล์คือในส่วนที่เป็นดาต้า แต่ที่สำคัญคือดอสจะเรียกเนื้อที่ในดิสค์ที่จัดสรรให้กับไฟล์ใน FAT คืน การกู้ไฟล์ในดอสจึงไม่ง่ายเหมือนในแอปเปิ้ลดอส เพราะเราขาดพอยน์เตอร์ที่จะติดตาม

2Eh ตามด้วย 2Eh คือจุดจุด ดังที่เห็นในไดเรคทอรีย่อย แสดงว่าเป็นเซกเตอร์เริ่มต้นของไดเรคทอรีย่อย จะกล่าวในหัวข้อไดเรคทอรีย่อย

นามสกุล (Extension) จำนวน 3 ไบต์ ตามข้อกำหนดของดอส ซึ่งในไดเรคทอรีจะไม่เก็บจุดไว้ด้วย

แอตทริบิวต์ (Attibute) 1 ไบต์ แสดงลักษณะเฉพาะของไฟล์ คือ ถ้าบิตใดเป็น 1 จะแสดงความหมายดังนี้

บิต 0 = Read Only เขียนทับไม่ได้

บิต 1 = Hidden เป็นไฟล์ที่ซ่อนไว้ ไม่แสดงชื่อไฟล์นี้เมื่อใช้คำสั่ง dir

บิต 2 = System เป็นไฟล์ในระบบจัดการคือ IBMBIO.COM และ IBM DOS.COM

บิต 3 = Label แสดงว่าชื่อในเอนทรีนี้เป็นชื่อ Volume

บิต 4 = Subdirectory แสดงว่าชื่อในเอนทรีนี้ เป็นชื่อไดเรคทอรีย่อย

บิต 5 = Archive ปกติจะเป็น 1 แตกต่างกับบิตอื่นที่กล่าวมา ใช้ในฮาร์ดดิสค์ เมื่อ BACK UP จะถูกทำให้เป็น 0 จนกว่าจะมีการเปลี่ยนแปลงแก้ไขจึงจะถูกทำให้เป็น 1 ใหม่

เวลา (Time) จำนวน 2 ไบต์ เป็นเวลาที่เก็บข้อมูลลงไฟล์ครั้งสุดท้ายเก็บในแบบ **hhhhhmmmmmmssssss**

ในเมื่อ h คือชั่วโมงจำนวน 5 บิต เก็บได้ตั้งแต่ 0 ถึง 31 ใช้ถึง 23

m คือนาทีจำนวน 6 บิต เก็บได้ตั้งแต่ 0 ถึง 64 ใช้ถึง 59

s คือวินาทีจำนวน 5 บิต เก็บได้ตั้งแต่ 0 ถึง 31 ใช้ถึง 29 ค่าวินาทีที่เก็บจึงเป็นค่าวินาทีหาร 2

วัน (date) จำนวน 2 ไบต์ เป็น ว.ด.ป. ที่เก็บข้อมูลลงไฟล์นี้ครั้งสุดท้าย เก็บในแบบ

yyyyyyymmmmddddd

Field	Offset	Description	Size (bytes)	Format
1	0	Filename	8	ASCII characters
2	8	Filename extension	3	ASCII characters
3	11	Attribute	1	Bit coded
4	12	Reserved	10	Unused; zeros
5	22	Time	2	Word, coded
6	24	Date	2	Word, coded
7	26	Starting FAT entry	2	Word
8	28	File size	4	Integer

รูปที่ 1 ข้อมูลที่เก็บอยู่ใน 1 directory entry ของดอสซึ่งอยู่ในส่วนที่เรียกว่าไดเรคทอรีบนแผ่นดิสค์

ในเมื่อ Y เป็นปีจำนวน 7 บิต เก็บได้ตั้งแต่ 0 ถึง 128 ซึ่งเป็นค่าที่ต้องนำไปบวกกับ ค.ศ. 1980 จึงเป็นค่าตั้งแต่ ค.ศ. 1980 ถึง 2107 แต่ดอสจะดำเนินการให้ถึง ค.ศ. 2099 เท่านั้น

m เป็นเดือนจำนวน 4 บิต เก็บได้ตั้งแต่ 0 ถึง 15 ใช้จริง 1 ถึง 12

d เป็นวันที่จำนวน 5 บิต เก็บได้ตั้งแต่ 0 ถึง 32 ใช้จริง 1 ถึง 31

คลัสเตอร์แรก (Starting FAT entry) จำนวน 2 ไบต์ เป็นพอยน์เตอร์ไปยังคลัสเตอร์แรกที่เก็บข้อมูลของไฟล์นี้ ให้นำมาคูณด้วย 2 แล้วบวกด้วย 8 มาจากเซกเตอร์ = (คลัสเตอร์- 2)×2+12 เพราะคลัสเตอร์แรกจะขึ้นต้นด้วยเลข 2 ตรงกับเซกเตอร์ 12

ขนาดของไฟล์ (File size) จำนวน 4 ไบต์ เก็บจำนวนไบต์ที่แท้จริงของไฟล์นั้น แต่สำหรับไฟล์ที่อ่านและเขียนเป็นบล็อก เช่น ไฟล์ที่สร้างด้วยโปรแกรมประเภทเวิร์ดโปรเซสซิ่ง ค่าละเอียดจะเป็น 128 ไบต์

ไดเรคทอรีย่อย (subdirectory)

ตัวชื่อไดเรคทอรีย่อยจะถูกเก็บอยู่ในไดเรคทอรีหลัก (root directory) คือที่กล่าวมา เช่นเดียวกับชื่อไฟล์ แต่ดอสจะไปเปิดไดเรคทอรีย่อยขึ้นที่เซกเตอร์หนึ่ง สำหรับเก็บชื่อไฟล์ที่อยู่ในไดเรคทอรีย่อย เช่นเดียวกับที่เก็บในไดเรคทอรีหลัก แต่จะขึ้นต้นไดเรคทอรีย่อยนี้ด้วยชื่อไฟล์เป็นจุดและจุดจุด

ทดลองอ่านไฟล์

เพื่อความเข้าใจเพิ่มผมจะฟอร์แมตดิสค์ในแบบมีซิสเท็ม แล้วสร้างไฟล์ชื่อ ONE.PAS แล้วสร้างไดเรคทอรีย่อยชื่อ SUBDIR1 ในไดเรคทอรีย่อยจะสร้างไฟล์ FIRST.TXT กับ SECOND.TXT เสร็จแล้วในไดเรคทอรีหลักสร้างไฟล์ TWO.PAS กับ THREE.PAS แล้วลบไฟล์ TWO.PAS ผมจะได้ไฟล์ในไดเรคทอรีหลักและย่อยดังในรูปที่ 2

จากนั้นผมใช้โปรแกรม DEBUG อ่านไดเรคทอรีจากดิสค์แผ่นนี้ ซึ่งผมใส่ไว้ในไดรฟ์ B มาลงหน่วยความจำที่แอดเดรส CS:100 (เพราะไม่มีโปรแกรมใดอยู่) จะได้ข้อมูลในไดเรคทอรีดังรูปที่ 3 ได้เขียนคำอธิบายบางเรื่องไว้ได้รูปกรุณาอ่านด้วยครับ

```
B>erase two.pas

B>dir

 Volume in drive B has no label
 Directory of  B:\

COMMAND  COM    17792  10-20-83   12:00p
ONE      PAS       92   3-20-87    5:12p
SUBDIR1      <DIR>      3-20-87    5:20p
THREE    PAS       96   3-20-87    5:16p
       4 File(s)    316416 bytes free

B>dir subdir1

 Volume in drive B has no label
 Directory of  B:\subdir1

.            <DIR>      3-20-87    5:20p
..           <DIR>      3-20-87    5:20p
FIRST    TXT       58   3-20-87    5:18p
SECOND   TXT       64   3-20-87    5:19p
       4 File(s)    316416 bytes free

B>
```

รูปที่ 2 แสดงชื่อไฟล์ในไดเรคทอรีหลักและไดเรคทอรีย่อย

```
D>debug
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=21EB  ES=21EB  SS=21EB  CS=21EB  IP=0100   NV UP DI PL NZ NA PO NC
21EB:0100 DAE8          ESC     15,AL
-l cs:100 1 5 1
-d cs:100 1ff
21EB:0100  49 42 4D 42 49 4F 20 20-43 4F 4D 27 00 00 00 00   IBMBIO  COM'....
21EB:0110  00 00 00 00 00 00 00 60-54 07 02 00 B0 12 00 00   .......'T.......
21EB:0120  49 42 4D 44 4F 53 20 20-43 4F 4D 27 00 00 00 00   IBMDOS  COM'....
21EB:0130  00 00 00 00 00 00 00 60-54 07 07 00 80 42 00 00   .......'T....B..
21EB:0140  43 4F 4D 4D 41 4E 44 20-43 4F 4D 20 00 00 00 00   COMMAND COM ....
21EB:0150  00 00 00 00 00 00 00 60-54 07 18 00 80 45 00 00   .......'T....E..
21EB:0160  4F 4E 45 20 20 20 20 20-50 41 53 20 00 00 00 00   ONE     PAS ....
21EB:0170  00 00 00 00 00 00 93 89-74 0E 2A 00 5C 00 00 00   ........t.*.\...
21EB:0180  53 55 42 44 49 52 31 20-20 20 10 00 00 00 00 00   SUBDIR1     .....
21EB:0190  00 00 00 00 00 00 8C 8A-74 0E 2B 00 00 00 00 00   ........t.+.....
21EB:01A0  E5 57 4F 20 20 20 20 20-50 41 53 20 00 00 00 00   eWO     PAS ....
21EB:01B0  00 00 00 00 00 00 DD 89-74 0E 2E 00 5D 00 00 00   ......].t...]...
21EB:01C0  54 48 52 45 45 20 20 20-50 41 53 20 00 00 00 00   THREE   PAS ....
21EB:01D0  00 00 00 00 00 00 10 8A-74 0E 2F 00 60 00 00 00   ........t./.'...
21EB:01E0  00 F6 F6 F6 F6 F6 F6 F6-F6 F6 F6 F6 F6 F6 F6 F6   .vvvvvvvvvvvvvvv
21EB:01F0  F6 F6 F6 F6 F6 F6 F6 F6-F6 F6 F6 F6 F6 F6 F6 F6   vvvvvvvvvvvvvvvv
```

รูปที่ 3 แสดงการใช้ DEBUG โหลดโปรแกรมจากดิสค์มาลงหน่วยความจำ จะเห็นว่าแอตทริบิวต์ของไฟล์ IBMBIO.COM และ IBMDOS.COM ซึ่งอยู่ต่อท้ายอักษร M (4 Dh) มีค่าเป็น 27 หมายถึงเป็นไฟล์ที่เขียนทับไม่ได้ เป็นไฟล์ที่ซ่อนอยู่ และเป็นไฟล์ที่เป็นซิสเต็มไฟล์ที่เป็นไดเรคทอรีย่อยจะมีค่าแอตทริบิวต์เป็น 10 h ส่วนไฟล์ธรรมดามีค่าแอตทริบิวต์เป็น 20 h ไฟล์ที่ถูกลบคือ TWO.PAS ถูกกำหนดให้ไบต์แรกเป็น E5h

จากชื่อไดเรคทอรีย่อย SUBDIR1 จะเห็นว่าพอยน์เตอร์ชี้ไปที่คลัสเตอร์ 002Bh นำมาคูณด้วย 2 แล้วบวกด้วย 8 จะได้เป็นเซกเตอร์ 5Eh ซึ่งเป็นที่เก็บไดเรคทอรีย่อย ผมได้โหลดลงหน่วยความจำอีก แล้วแสดงให้ดูในรูปที่ 4 จากนี้ผมดูในชื่อไฟล์ FIRST.TXT ได้พอยน์เตอร์ไปยังคลัสเตอร์ 002C คือเซกเตอร์ 60h โหลดข้อมูลมาให้ดูดังรูปที่ 5 ขอให้เทียบกับข้อมูลในไฟล์นี้ซึ่งผมให้พิมพ์ไว้ตรงส่วนล่างของรูปนี้

จากการทดลองนี้ ผมคิดว่าคงจะพอมองออกนะครับว่า ดอสเก็บไดเรคทอรีหลักอย่างไร เก็บไดเรคทอรีย่อยอย่างไร และเก็บข้อมูลในไฟล์อย่างไร

เรากับดอส

โดยปกติการติดต่อกับดิสค์เราจะดำเนินการผ่านดอส บางท่านอาจสงสัยว่าถ้าไม่ปกติเป็นอย่างไร ถ้าไม่ปกติก็คือการที่เราดำเนินการผ่าน BIOS หรือไม่ก็เขียนโปรแกรมดำเนินการกับดิสค์ไดรฟ์โดยตรง เท่ากับว่าเราไม่ใช้สิ่งอำนวยความสะดวกที่ดอสจัดหาไว้ให้ ในการติดต่อกับดิสค์นั้น ดอสกำหนดให้เราติดต่อในนามของไฟล์ ซึ่งเราจะต้องเตรียมการในบางเรื่อง

ผมขอเรียนให้ทราบเสียก่อนว่าเรื่องที่ผมจะกล่าวถึงต่อไปนี้ไม่ค่อยจำเป็นนักสำหรับดอสเวอร์ชัน 2.0 ขึ้นมา มีกรณีที่ใช้น้อยมาก ดังเช่นที่ผมนำมาใช้ใน *"ปาสคาลอรรถกร"* ฉบับนี้ ทั้งนี้เพราะในดอสเวอร์ชัน 1.x การติดต่อกับไฟล์ทำในแบบ sequential หรือ random หรือ random block ซึ่งพัฒนามาจาก CP/M 80 แต่ตั้งแต่เวอร์ชัน 2.0 เป็นต้นมาดอสใช้กรรมวิธีที่เรียกว่า file handling ดังจะกล่าวในตอนท้าย

เมื่อเราสั่งรันโปรแกรมหนึ่งโปรแกรมใดนั้น ดอสจะโหลดโปรแกรมนั้นลงหน่วยความจำก่อนที่จะรัน แต่ก่อนที่จะโหลดโปรแกรมดังกล่าว ดอสจะเตรียมหน่วยความจำส่วนหนึ่งขนาด 256 ไบต์ เพื่อใช้สำหรับการดำเนินการต่างๆ เรียกว่า **PSP** (Program Segment Prefix) แล้วดำเนินการดังนี้

- ถ้าเป็นไฟล์ .COM จะให้เซกเมนต์รีจิสเตอร์ทุกตัวชี้ไปที่ต้น PSP โดยมีค่าออฟเซ็ตของต้น PSP เป็น 0000h และให้ IP มีค่าเป็น 0100h แล้วโหลดโปรแกรมลงที่ CS:IP
- ถ้าเป็นไฟล์ .EXE จะให้ DS และ ES ชี้ที่ต้น PSP ดังกล่าว ให้ CS มีค่าเป็น DS+10h (หรือมากกว่านี้ขึ้นอยู่กับรูปแบบของโปรแกรม .EXE และการลิงค์) ให้ IP มีค่าเป็น 0000h แล้วโหลดโปรแกรมลงที่ CS:IP นั่นคือโดยปกติจะโหลดโปรแกรมไม่ว่าจะเป็น .COM หรือ .EXE ลงตำแหน่งเดียวกัน เพราะค่า 10h ที่บวกให้แก่ CS กับค่า 100h ที่ให้แก่ IP คือค่าเดียวกัน

ในส่วนที่เป็น PSP ประกอบด้วย

```
-l cs:100 1 5e 2
-d cs:100 1ff
21EB:0100  2E 20 20 20 20 20 20 20-20 20 20 10 00 00 00 00   .          .....
21EB:0110  00 00 00 00 00 00 8C 8A-74 0E 2B 00 00 00 00 00   ........t.+.....
21EB:0120  2E 2E 20 20 20 20 20 20-20 20 20 10 00 00 00 00   ..         .....
21EB:0130  00 00 00 00 00 00 8C 8A-74 0E 00 00 00 00 00 00   ........t.......
21EB:0140  46 49 52 53 54 20 20 20-54 58 54 20 00 00 00 00   FIRST   TXT ....
21EB:0150  00 00 00 00 00 00 4B 8A-74 0E 2C 00 3A 00 00 00   ......K.t.,.:...
21EB:0160  53 45 43 4F 4E 44 20 20-54 58 54 20 00 00 00 00   SECOND  TXT ....
21EB:0170  00 00 00 00 00 00 72 8A-74 0E 2D 00 40 00 00 00   ......r.t.-.@...
21EB:0180  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:0190  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01A0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01B0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01C0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01D0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01E0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
21EB:01F0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

รูปที่ 4 แสดงไดเรคทอรีของไดเรคทอรีย่อย

```
-l cs:100 1 60 1
-d cs:100 1ff
21EB:0100  54 68 69 73 20 69 73 20-46 49 52 53 54 20 54 45   This is FIRST TE
21EB:0110  58 54 0D 0A 54 68 69 73-20 74 65 78 74 20 69 73   XT..This text is
21EB:0120  20 69 6E 20 73 75 62 64-69 72 65 63 74 6F 72 79    in subdirectory
21EB:0130  20 53 55 42 44 49 52 31-2E 1A 00 00 00 00 00 00    SUBDIR1.........
21EB:0140  46 49 52 53 54 20 20 20-54 58 54 00 00 00 00 00   FIRST   TXT.....
21EB:0150  00 00 00 00 00 00 AD 8A-74 0E 00 00 00 00 00 00   ......-.t.......
21EB:0160  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................

B>type subdir1\first.txt
This is FIRST TEXT
This text is in subdirectory SUBDIR1.
B>
```

รูปที่ 5 แสดงข้อมูลที่เก็บอยู่ในไฟล์ FIRST.TXT

ส่วนต่างๆ ดังแสดงในรูปที่ 6 ซึ่งสามารถรวมเป็นส่วนใหญ่ๆ ได้ 5 ส่วน ดังนี้คือ

ส่วนอ้างอิง

คือ ตั้งแต่ฟิลด์ 1 ถึงฟิลด์ 11 เป็นส่วนที่ดอสใช้ในการอ้างอิงต่างๆ ที่น่าสนใจคือฟิลด์ 0 เก็บค่าเป็น CD 20 คือ INT 20h เพื่อจบโปรแกรมกลับไปยังดอส บางท่านจึงนิยมเขียนโปรแกรมโดยขึ้นต้นและลงท้ายว่า

```
PUSH DS; คือต้น PSP
MOV  AX, 0
PUSH AX
.
.
RETF
```

ซึ่งจะทำให้ปฏิบัติ คำสั่ง INT 20h หลัง RETF แล้ว แต่วิธีนี้ไม่ใช่วิธีที่ดี เพราะ INT 20h จะไม่ปิดไฟล์ ดอสสร้างไว้เพื่อให้โปรแกรม LINK ใช้

Parameter Area ฟิลด์ 17 และ 18

ผมขอข้ามมาส่วนนี้ก่อน เพราะจะเป็นส่วนที่สัมพันธ์กับ FCB ส่วนนี้จะเป็นส่วนที่เก็บข้อความในคอมมานด์-ไลน์ตามที่ผู้ใช้ป้อน เช่น

A > rdecomln B: one.pas

โดยที่ให้ rdecomln เป็นโปรแกรมที่จะดำเนินการกับไฟล์ที่เป็นเท็กซ์ไฟล์ ดอสจะนับจำนวนอักขระในบรรทัดนี้ทั้งหมดที่อยู่หลังคำว่า rdecomln อักขระแรก คือวรรคในกรณีนี้จะนับได้ 10 ไบต์ นำไปใช้ที่ไบต์แรกของ Parameter Area คือไบต์ที่ 80h ของ PSP และต่อด้วยข้อความดังกล่าวทั้งหมด ได้ทดลองให้ดูโดยใช้ DEBUG ดังแสดงในรูปที่ 7

จะเห็นว่าดอสได้อ่านข้อความในคอมมานด์ไลน์ทั้งหมด คือข้อความหลังเครื่องหมาย > มาไว้ในส่วนที่เป็นพารา-มิเตอร์คือที่แอดเดรส DS:80 ถึง DS:9B (เพราะปิดท้ายด้วย 0Dh) ก่อน แล้วยกเลิกชื่อไฟล์ที่จะรัน แล้วเลื่อนชื่อไฟล์ต่อท้ายมาไว้เริ่มต้นที่ DS:81 แล้วให้ความยาวที่ DS:80 เป็น 0Ah คือ 10 จะเห็นว่ามีร่องรอยของชื่อไฟล์ที่รันคือคำว่า COM อยู่

การที่ดอสทำเช่นนี้ทำให้โปรแกรมที่กำลังรันอยู่ไม่ทราบว่าตนเองชื่ออะไร (คือเราไม่สามารถหาชื่อไฟล์ของโปรแกรมที่กำลังรันอยู่) นอกจากนี้ดอสไม่เก็บเครื่องหมายมากกว่า (>) หรือน้อยกว่า (<) ซึ่งเป็นการเปลี่ยนทิศทางของอินพุตและเอาต์พุต ทำให้ไม่สามารถตรวจสอบว่าขณะนั้นโปรแกรมกำลังติดต่ออยู่กับอุปกรณ์ใด

FCB (file control block) ฟิลด์ 13 ถึง 16

เป็นการใช้ไฟล์อย่างแบบเก่า (คือ

Field	Offset Dec	Offset Hex	Length	Description
1	0	0	2	INT 32 instruction
2	2	2	2	Size of memory (in paragraphs)
3	4	4	1	Reserved; normally 0
4	5	5	5	Call to DOS function dispatcher
5	10	A	4	Terminate vector
6	14	E	4	Break vector
7	18	12	4	Error vector
8	22	16	22	Used by DOS
9	44	2C	2	Environment strings pointer
10	46	2E	34	DOS work area
11	80	50	3	INT 33, RETF instructions
12	83	53	2	Reserved
13	85	55	7	FCB #1 extension
14	92	5C	9	FCB #1
15	101	65	7	FCB #2 extension
16	108	6C	20	FCB #2
17	128	80	1	Parameter length
18	129	81	127	Parameters
19	128	80	128	Disk transfer area (DTA)

รูปที่ 6 แสดงข้อมูลที่เก็บใน PSP จะเห็นว่าฟิลด์ที่ 19 ทับกับฟิลด์ 17 กับ 18

```
D>debug b:rdecomln.com b:one.pas
-r
AX=0000  BX=0000  CX=2CD2  DX=0000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0919  ES=0919  SS=0919  CS=0919  IP=0100   NV UP DI PL NZ NA PO NC
0919:0100 E9792C        JMP     2D7C
-d ds:50 ff
0919:0050  CD 21 CB 00 00 00 00 00-00 00 00 00 02 4F 4E 45   M!K..........ONE
0919:0060  20 20 20 20 20 50 41 53-00 00 00 00 00 20 20 20        PAS.....
0919:0070  20 20 20 20 20 20 20 20-00 00 00 00 00 00 00 00              ........
0919:0080  0A 20 62 3A 6F 6E 65 2E-70 61 73 0D 2E 63 6F 6D   . b:one.pas..com
0919:0090  20 62 3A 6F 6E 65 2E 70-61 73 0D 00 00 00 00 00    b:one.pas......
0919:00A0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0919:00B0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

รูปที่ 7 แสดงการเก็บค่าในคอมมานด์ไลน์ไว้ใน Parameter Area

เมื่อมีการอ่านเขียนกับไฟล์ในแบบ sequential, random และ random block) ค่าที่เก็บใน FCB จะเป็นดังแสดงในรูปที่ 8 ขอให้ดูประกอบกับรูปที่ 6 ครับ ในรูปที่ 8 ค่าที่ให้เป็นค่าออฟเซ็ต ถ้าเป็น FCB#1 ออฟเซ็ต- 7 จะอยู่ที่ 55h ตัว FCB แท้ๆเริ่มที่ 5Ch คือตรงออฟเซ็ต 0 จะเป็นหมายเลขไดรฟ์ของไฟล์ แต่ให้ไดรฟ์ A มีค่าเป็น 1 ต่อด้วยชื่อไฟล์

ขอให้ดูในรูปที่ 7 ชื่อไฟล์ที่ป้อนในคอมมานด์ไลน์ และถูกเก็บในพารามิเตอร์ดังกล่าวแล้ว จะถูกนำมาไว้ใน FCB ขอให้สังเกตด้วยว่าเนื่องจากชื่อไฟล์มีเพียง 8 ไบต์ ดอสจะเติม space คือ 20h อีก 5 ไบต์ และหมายเลขไดรฟ์เป็น 02h อยู่หน้าชื่อไฟล์ สำหรับเรื่องราวของ FCB อื่นๆ ของดกล่าวครับ เพราะเลิกใช้แล้ว

DTA (Data Transfer Area)

ในการอ่านและเขียนกับไฟล์ในแบบ Sequential, Random และ Random block ดอสจะดำเนินการอ่านหรือเขียนทีละ 1 เรคอร์ด คือครั้งละ 128 ไบต์ โดยอ่านจากดิสค์มาลงตรง DTA ก่อน และในทางตรงข้ามคือ นำข้อมูลใน DTA ไปเขียนลงดิสค์ เป็นหน้าที่ของผู้เขียนโปรแกรมที่จะเคลื่อนย้ายข้อมูลไปดำเนินการ หรือนำมาให้แก่ DTA ก่อนที่จะอ่านและเขียนกับดิสค์ในครั้งต่อไป

โดยทั่วไปถือว่า DTA เป็นส่วนหนึ่งของ PSP โดยนับเป็นฟิลด์ที่ 19 อยู่ที่ไบต์ 80h ถึง FFh คือซ้อนทับกับส่วนที่เป็น Parameter Area เรื่องการซ้อนทับไม่เป็นปัญหา เพราะจะใช้ส่วนความจำส่วนนี้ในฐานะ Parameter Area เมื่อลอกข้อความจากคอมมานด์ไลน์ และจะใช้เป็น DTA เมื่อติดต่อกับดิสค์ แต่ดอสก็ให้เราสามารถย้าย DTA ออกจาก PSP ไปไว้ที่ใดก็ได้ตามต้องการ คือเราสามารถกำหนดหน่วยความจำ ณ ที่ใดก็ได้ให้เป็น DTA โดยใช้ฟังก์ชัน 1Ah (set DTA)

ของใหม่ของดอส

นับตั้งแต่ดอสเวอร์ชัน 2.0 เป็นต้นมา คือตั้งแต่ฟังก์ชันดอส 2Fh และที่มากกว่าดอสได้เปลี่ยนแนวความคิดในการติดต่อกับดิสค์ใหม่ แต่ขอเรียนให้ทราบถึงการติดต่อกับดิสค์ก่อนว่า เราสามารถแยกการติดต่อกับดิสค์ได้เป็น 2 ลักษณะคือ เรื่องแรกเป็นเรื่องที่เกี่ยวกับไดเรคทอรี เช่น การ delete, rename และอื่นๆ อีกเรื่องหนึ่งเป็นเรื่องที่เกี่ยวกับการอ่านและเขียนกับไฟล์

ในการเปลี่ยนแปลงนี้ PSP ยังคงมีอยู่ และดอสอาจจะใช้เป็นการภายในของดอส โดยเฉพาะในเรื่องที่เกี่ยวกับไดเรคทอรี แต่สำหรับการอ่านและเขียนกับไฟล์ ได้เปลี่ยนไปโดยสิ้นเชิง คือ

1. **ใช้ ASCIIZ แทน FCB** ซึ่ง ASCIIZ คือสตริงใดก็ได้ที่ปิดท้ายด้วยอักขระ NULL คือ 00h หรือ 0 เมื่อเป็นชื่อไฟล์ก็จะเป็นสตริงของ file path เช่น **'b: subdir1\subdir2\income.txt', 00h** ทั้งนี้เพราะในดอส 1.x ไม่มีไดเรคทอรีย่อย และ FCB อ้างถึงไดเรคทอรีย่อยไม่ได้ ในการใช้งานจะให้ DS:DX ชี้ที่ ASCIIZ

2. **ใช้** File Handles **แทน FCB** เมื่อใช้ FCB นั้นเมื่อเปลี่ยนไฟล์ ต้องย้ายชื่อไฟล์ใน FCB ไปและมา แต่สำหรับในแบบใหม่นี้ เมื่อเราเปิดไฟล์ดอสจะให้หมายเลข เรียกว่า File Handle กลับคืนมาในรีจิสเตอร์ AX ซึ่งเราต้องเก็บไว้เพื่ออ้างอิงเมื่อจะใช้ไฟล์นี้ต่อไป เช่น ในการอ่านและเขียน ทำให้สะดวกมากกว่าใช้ FCB

3. **ใช้ buffer แทน DTA** ถึงแม้ในหนังสือที่เกี่ยวกับดอสบางเล่มยังคงเรียกว่า DTA แต่ความจริงแล้วไม่ใช่ เพราะเราไม่ต้องเซตให้เป็น DTA การใช้งานเพียงแต่ให้ DS:DX ชี้มาที่ต้นบัฟเฟอร์นี้เท่านั้น

4. **อ่าน-เขียนเป็นไบต์ ตามจำนวนไบต์ที่ต้องการ** แทนที่จะเป็นครั้งละ 1 เรคอร์ด (128 ไบต์) ทำให้สามารถดำเนินการกับข้อมูลที่เป็นโครงสร้างได้โดยสะดวก และกำหนดขนาดของบัฟเฟอร์ได้ตามความเหมาะสม

ผมหวังว่าเท่าที่กล่าวมานี้ คงทำให้ทราบเรื่องราวของดอสอย่างกว้าง ๆ และทราบว่าเรื่องใดจำเป็นต้องศึกษา เรื่องใดไม่ต้อง ฉบับหน้าจะเป็นเรื่องราวของภาษาเครื่องต่อไป และจะเป็นตอนจบ

Offset (dec)	Size (bytes)	Set by	Description
−7	1	Us-1	Extension active signal: FF = yes, otherwise no
−6	5	Us-1	Nominally unused: should be set to zeros
−1	1	Us-1	File attribute, when extension active
0	1	Us-1	Special drive number
1	8	Us-1	Filename or device name
9	3	Us-1	Filename extension
12	2	Us-2	Current block number
14	2	Us-2	Record size
16	4	DOS	File size in bytes
20	2	DOS	File date (bit-coded, as in file directory)
22	10	DOS	Miscellaneous DOS control work area
32	1	Us-2	Current record number ("signed" byte, see text)
33	4	Us-2	Random record number

รูปที่ 8 ข้อมูลที่เก็บใน FCB